NASA-CR-200292

[illegible]

[illegible]

FINAL REPORT TO
NASA AMES RESEARCH CENTER

NASA AWARD NUMBER NCC 2-538
COVERING THE PERIOD
JANUARY 1, 1988 - MAY 31, 1995

"RESEARCH ON SOAR", "LEARNING TO USE DEVICES", AND "INDUCTION AS KNOWLEDGE INTEGRATION"

Paul S. Rosenbloom (Principal Investigator)

INFORMATION SCIENCES INSTITUTE
UNIVERSITY OF SOUTHERN CALIFORNIA
LOS ANGELES, CALIFORNIA 90089-1147

March 1996

Work on this effort divided up into three phases. The first phase, which spanned the first three years of the effort, focused on "Research on Soar". The second phase, which spanned the next 2.5 years of the effort, focused on "Learning to Use Devices" in the context of Soar. The third phase, which spanned the last two years of the effort, focused on "Induction as Knowledge Integration". These three phases will be described in brief in the following three sections.

1. Research on Soar

The first phase covered research on intelligent behavior in the context of the Soar architecture (Laird, Newell, & Rosenbloom, 1987; Rosenbloom *et al*, 1991; Rosenbloom, Laird, & Newell, 1993). Two threads focused on learning – one on the acquisition of new knowledge (*knowledge level learning*) and the other on the reformulation of existing knowledge so as to improve performance (*symbol level learning*). The third thread focused on analogy/case-based-reasoning. The fourth and final thread focused on the relationship between Soar and Connectionism.

1.1. Knowledge Level Learning

Although Soar's learning mechanism was called *chunking* based on a related psychological notion of the same name (Miller, 1956), it gradually became apparent that Soar's chunking mechanism had a great deal of difficulty producing the kind of declarative learning – that is, the acquisition of new knowledge from the outside – that had become so central to the concept in psychology. Instead, Soar's chunking appeared to be limited to pure speed up learning. Prior to this effort we had begun to lay out how Soar's chunking could in fact acquire new knowledge. As part of this effort, this capability was better understood, implemented in a general fashion, and extended to handle simple forms of inductive learning (Rosenbloom & Aasman, 1990; Rosenbloom, Newell, & Laird, 1991). It also led to the development of a new polynomially bounded generalization algorithm for version spaces (Smith & Rosenbloom, 1990), and was one of the key inputs in developing the ideas that led to the third phase of this effort (Section 3).

1.2. Symbol Level Learning

The focus of this effort was on understanding the problem of expensive chunks and on investigating the space of possible solutions to the problem. Chunking is a learning mechanism that acquires new rules from traces of system problem solving. These new rules are (usually) intended to speed up the system via a form of caching, in which extended problem solving is replaced by the match and firing of previously learned rules. Unfortunately, the match process for new rules can be exponential (in the number of rule conditions) in the worst case, so that sometimes the acquisition of new rules actually slows down the system rather than speeding it up. This is the phenomenon of expensive chunks (Tambe, Newell, & Rosenbloom, 1990), which is a form of the utility problem in explanation based learning (Minton, 1988).

The identification of the cause of expensive chunks as being the presence of *multi-attributes* – attributes of objects that can simultaneously take on more than one value – led to the identification of a restriction on the system's expressiveness that can provide a linear-time guarantee for the match of individual rules (Tambe & Rosenbloom, 1988; Tambe & Rosenbloom, 1989; Tambe, Newell, & Rosenbloom, 1990). This *unique-attribute* restriction ensures that every attribute of every object has at most one value at a time. It eliminates slowdowns, but at the cost of making it more difficult to express certain kinds of structures – basically, undifferentiated sets.

Follow-on work further investigated the space of possible solutions to the problem of expensive chunks, with an eye out for alternatives that traded-off less in the way of expressibility. First, four dimensions were identified in terms of local syntactic constraints on working memory – number of attributes per object, number of values per attribute, number of attributes per value, and number of objects per attribute – whose cross-product defines a space of possible alternatives. Within this space, it was possible to prove that unique-attributes was the best alternative (Tambe & Rosenbloom, 1990; Tambe & Rosenbloom, 1994).

Subsequently a second space of alternatives was discovered in which alternatives are defined according to different ways of bounding the search performed during the match process. The one alternative that was investigated in some depth was *instantiationless match*, in which instead of keeping track of which bindings of which variables go together, it is only possible to keep track of each variables set of bindings separately (Tambe & Rosenbloom, 1994). Instantiationless match reduces the cost of learned rules by eliminating the cross-product effects that occur when it is necessary to keep track of which elements of each variable's set of bindings go together.

While instantiationless match has not been turned into as practical an approach as has unique-attributes, it did succeed in creating an important tie between production match and constraint satisfaction. It also has led to a radically new match algorithm, called *collection match* (Acharya & Tambe, 1993). Although collection match does not provide the kinds of complexity bounds needed to eliminate expensive chunks – and was not investigated as part of this effort – it does appear to provide a highly efficient and practical match algorithm for the standard production match problem. In particular, it scales particularly well as the size of working memory grows.

1.3. Combining Rule-Based and Case-Based Reasoning

This effort started with a focus on advice taking and analogical reasoning within Soar, but then evolved into a study outside of Soar of the combination of rule-based reasoning (RBR) and case-based reasoning (CBR) in the domain of proper name pronunciation. The basic idea is to use general – but only approximately correct – rules to suggest alternatives, and then to search a case base for compelling exceptions to the rules (Golding, 1991; Golding & Rosenbloom, 1991; Golding & Rosenbloom, 1996). This allows these two distinct sources of knowledge to be used synergistically to improve overall accuracy, and in so doing allows a lightly engineered academic system for Proper Name Pronunciation to be competitive with existing heavily engineered commercial systems (Golding & Rosenbloom, 1993). This basic architecture should be usable beyond the domain of Proper Name Pronunciation, whenever reasonably accurate and efficient – but not perfect – rules are available, along with a set of relevant cases and a similarity metric (for evaluating case closeness).

Other innovations in the work include using the rule base in service of case indexing (*prediction-based indexing*) and adaptation (*rational reconstruction*), so that improvements in the rules lead to faster and better CBR; the use of positive analogies to help choose between plausible rule-proposed alternatives that are not eliminated by negative analogies (i.e., by exceptions); the identification of the phenomenon of ***analogical decline***, in which fewer good analogies are found for rarer names; a method for automatically extending an incomplete domain theory; a method for evaluating proposed analogies via their accuracy over the entire case base; and an automated threshold setting procedure.

Two of the papers on this work won prizes. Golding & Rosenbloom (1993) won the the American Voice Input/Output Society Gary K. Poock Editor's Award for the Outstanding Paper in the *AVIOS Journal* during 1993. Golding & Rosenbloom (1991) won the award for the best written paper at the Ninth National Conference on Artificial Intelligence.

1.4. Connectionism, Goal-Oriented Behavior, and Soar

This effort performed a functional analysis of goal-oriented behavior; used it to analyze both Soar and Connectionism and to create a mapping between the two; and outlined a strategy for creating a hybrid Connectionist Soar (Rosenbloom, 1989). The most interesting outcome of this investigation was the realization that Soar's approach to memory access – which is based on iterations of parallel rule firing until quiescence – maps quite closely onto the memory access approach typically used in connectionist networks. There are of course differences in the details, but this mapping lets us consider a Soar-like system that replaces Soar's memory structure with a connectionist net, yet still supports Soar's higher level goal-oriented capabilities. This idea was later followed up outside of this effort in a partial connectionist reimplementation of Soar, called Neuro-Soar (Cho, Rosenbloom, & Dolan, 1991).

2. Learning to Use Devices

When presented with a new device to use – such as a VCR, a data logger, a network configurer, or an experimental package – people are able to learn to use the device based on a combination of knowledge sources. Typical knowledge sources might be manuals, experimentation with the device, knowledge of related devices, and verbal instructions. The effort in this phase focused on providing the same kind of capability to automated systems, so that when associated devices are upgraded or changed, the system can learn how to use the new devices from its available sources of knowledge, rather than requiring reprogramming.

One outcome of this effort was the development of a system that can start out with no knowledge about a video cassette recorder (VCR) and learn to play a tape on the VCR based on a single knowledge source – analogy with an audio cassette deck (ACD). The system is constructed in Soar as a set of five problem spaces, where each problem space corresponds to a model of a device, or an interface, or a domain, or some other coherent body of knowledge. The particular problem spaces used here are: the top-space, which interacts with the world; two domain spaces that characterize classes of tasks in the world (listen-to-music and watch-a-movie); a device model for the ACD; and a communication model that knows which buttons to press to elicit a desired function from the ACD. The top space begins with only primitive motor commands and perceptual input. If it does not know what to do, it reaches an impasse, and uses another problem space (such as watch-a-movie) in acquiring a task operator. This generic process of using other spaces to generate information usable in performing the task continues until the task has been refined enough that motor commands can be executed that will accomplish it.

In addition to learning from an analogical model, we interfaced a version of NL-Soar – the Soar-based natural language understanding capability being developed at CMU (Lehman, Lewis, & Newell, 1991) – to the system that learns to play tapes on the VCR, and extended it to demonstrate the acquisition of a fragment of relevant VCR knowledge from manual-like natural-language input. In particular, it processed the sentence "To play a tape, press the PLAY button" to yield part of the VCR device knowledge. There is admittedly still a long way to go here, but there are some very interesting opportunities and issues that it raises, including how to use the system's models (of devices and domains) as much of the semantics of the natural-language comprehension system, and how to use other knowledge sources (such as models of similar devices) to help in understanding and disambiguating the manual.[1]

In addition to the VCR domain, a system was also developed that could start out with no knowledge of a text editor, and learn to edit simple strings based on two knowledge sources: analogy with a typewriter, and observation of the effects of proposed actions on the text editor. This system uses two knowledge sources: a model of a related device and experimentation with, and observation of, the actual device.

The key intellectual issue that showed up in constructing these systems (outside of the the NL related issues) was how to perform the coupling between two arbitrary problem spaces that are used within a goal-subgoal relationship – even when the coupling hadn't been anticipated; that is, how to encode an impasse in one space into a problem to be solved in the other space, and how to encode the results of the solution in the subspace back into useful information in the parent space. To date in Soar this issue has always been addressed by hardcoding translation rules that create initial states in new problem spaces and return results to parent spaces. However, such an approach is inadequate when the range of impasses that can occur, and the range of spaces that can potentially be used for them, is quite large.

This problem can be viewed as a generalization of the analogy problem. In analogy, the two spaces are both device models, but about different devices (such as the VCR and ACD). Whereas here, in addition to this normal analogical relationship, there may be a need to map from a model of the domain to a model of the device, or between manual knowledge and a device space.

We ended up casting this problem in a comprehension-based framework. When an impasse occurs, and a problem space is selected for it in a subgoal, the subgoal space first comprehends the impassed space in its own terms – just as if, for example, it were an utterance in natural language – to determine how to formulate the problem in terms of similar tasks that it under-

[1]This approach bears a close relationship to a proposal by Goel and Eiselt on *model-based text interpretation and and knowledge acquisition* (Goel & Eiselt, 1991).

stands. This corresponds to a similar problem in the selected space. The similar problem is solved, and the solution returned to the parent space. The result is returned by proposing a comprehension operator for the parent-space that is to comprehend the solution. When the solution is comprehended, the parent space should have enough information to resolve the impasse. This comprehension process should have much in common structurally with other comprehension processes, such as natural language (NL) comprehension, but the knowledge used by it is likely to be different.

This phase of the effort was terminated at this point. Some interesting work had been done, and interesting issues and frameworks had been identified. However, it was still a long way from the kinds of results needed for the PhD thesis of which it was to be the basis. The basic problem was two-fold: (1) using in a general manner any one of the knowledge sources upon which learning was to be based was clearly (at least at this point) a massive undertaking; and (2) sufficient ideas for how to convert the comprehension-based framework for mapping into some form of general theory and implementation were lacking. As a consequence, we switched focus to the more well-defined topic of the third phase: induction as knowledge integration.

3. Induction as Knowledge Integration

The third phase arose out of the earlier work on knowledge level learning (Section 1.1); out of an earlier attempt to generate a general framework for the acquisition of new knowledge (Rosenbloom, 1988); and from analysis of a range of induction algorithms, in particular knowledge-intensive induction algorithms such as IVSM (Hirsh, 1990) and GRENDEL (Cohen, 1992). The goal was to develop a general means for integrating knowledge into induction. The key assumption motivating this was that the best way to improve both the accuracy and efficiency of induction algorithms is to use whatever knowledge might be available and relevant. Only once the knowledge is exhausted, does it make sense to try to tune the other aspects of the induction algorithm. In contrast, most of machine learning is focused on tuning induction algorithms that use examples as their only form of knowledge. The few systems that can use any knowledge beyond the examples, can generally use only a few specific forms of knowledge, and these in only a small number of predetermined ways.

Our investigations during this phase led us through three different frameworks for the integration of knowledge into induction. The *Transformational Framework* analyzed induction algorithms as black boxes with input ports for knowledge (Rosenbloom *et al*, 1993). Additional knowledge was integrated in by either reformulating the existing set of ports or by developing preprocessors that that translated the new knowledge into forms understandable by the existing ports; for example, converting constraints on the concept into pseudo-examples. The *Problem Space Framework* analyzes induction as search over a space of hypotheses (Rosenbloom *et al*, 1993). Although this is not particularly a new concept (Simon & Lea, 1974; Mitchell, 1979; Rosenbloom, 1988), the key new idea was to understand the use of knowledge in induction in terms of how it specifies, constrains, and orders the elements of this search space. The *Constraint Framework* is much like the Problem Space Framework; however, instead of focusing on search, it focuses on the integration of constraints and preferences over sets of hypotheses.

The constraint framework was developed sufficiently to form the basis for a PhD Thesis (Smith, 1995), with additional publications based on it also in progress. Here's the abstract of the thesis:

Accuracy and efficiency are the two main evaluation criteria for induction algorithms. One of the most powerful ways to improve performance along these dimensions is by integrating additional knowledge into the induction process. However, integrating knowledge that differs significantly from the knowledge already used by the algorithm usually requires rewriting the algorithm.

This dissertation presents KII, a Knowledge Integration framework for Induction, that provides a straightforward method for integrating knowledge into induction, and provides new insights into the effects of knowledge on the accuracy and complexity of induction. The idea behind KII is to express *all* knowledge uniformly as constraints and preferences on hypotheses. Knowledge is integrated by conjoining constraints and disjoining preferences. A hypothesis is induced from the integrated knowledge by finding a hypothesis consistent with all of the constraints and maximally preferred by the preferences.

Theoretically, just about any knowledge can be expressed in this manner. In practice, the constraint and preference languages determine both the knowledge that can be expressed and the complexity of identifying a consistent hypothesis. RS-KII, an instantiation of KII based on a very expressive set representation, is described. RS-KII can utilize the knowledge of at least two disparate induction algorithms – AQ-11 and CEA ("version spaces") – in addition to knowledge neither algorithm can utilize. It seems likely that RS-KII can utilize knowledge from other induction algorithms, as well as novel kinds of knowledge, but this is left for future work. RS-KII's complexity is comparable to these algorithms when using only the knowledge of a given algorithm, and in some cases RS-KII's complexity is dramatically superior. KII also provides new insights into the effects of knowledge on induction that are used to derive classes of knowledge for which induction is not computable.

4. References

Acharya, A., & Tambe, M. (1993). Collection-oriented match. *Proceedings of the Second International Conference on Information and Knowledge Management.* .

Cho, B., Rosenbloom, P. S., & Dolan, C. P. (1991). Neuro-Soar: A neural-network architecture for goal-oriented behavior. *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society.* Chicago, IL, Lawrence Erlbaum Associates.

Cohen, W. W. (1992). Compiling prior knowledge into an explicit bias. D. Sleeman & P. Edwards (Ed.), *Machine Learning: Proceedings of the Ninth International Workshop.* Aberdeen.

Goel, A, K. & Eiselt, K. P. (1991). Mental models, text interpretation, and knowledge acquisition. *SIGART Bulletin*, 2, 75-78.

Golding, A. R. (October 1991). *Pronouncing Names by a Combination of Rule-Based and Case-Based Reasoning.* Doctoral dissertation, Stanford University,

Golding, A., & Rosenbloom, P. S. (1991). Improving rule-based systems through case-based reasoning. *Proceedings of the Ninth National Conference on Artificial Intelligence.* Anaheim, CA: AAAI, MIT Press.

Golding, A. & Rosenbloom, P. S. (1993). A comparison of Anapron with seven other name-pronunciation systems. *Journal of the American Voice I/O Society, 14*, 1-21.

Golding, A. R., & Rosenbloom, P. S. (1996). Improving accuracy by combining rule-based and case-based reasoning. *Artificial Intelligence*, . In press.

Hirsh, H. (1990). *Incremental Version Space Merging: A General Framework for Concept Learning*. Boston, MA: Kluwer Academic Publishers.

Laird, J. E., Newell, A., & Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence, 33*, 1-64.

Lehman, J. F., Lewis, R. L., & Newell, A. (1991). Integrating knowledge sources in language comprehension. *Proceedings of the Thirteenth Annual Meeting of the Cognitive Science Society*. Hillsdale, NJ, Erlbaum.

Miller, G. A. (1956). The magic number seven plus or minus two: Some limits on our capacity for processing information. *Psychological Review, 63*, 81-97.

Minton, S. (1988). Quantitative results concerning the utility of explanation-based learning. *Proceedings of AAAI-88*. St. Paul, MN.

Mitchell, T. M. (1979). An Analysis of Generalization as a Search Problem. *Proceedings of IJCAI-79*. .

Rosenbloom, P. S. (1988). Beyond generalization as search: Towards a unified framework for the acquisition of new knowledge. G. F. DeJong (Ed.), *Proceedings of the AAAI Symposium on Explanation-Based Learning*. Stanford, CA: AAAI.

Rosenbloom, P. S. (1989). A symbolic goal-oriented perspective on connectionism and Soar. In R. Pfeifer, Z. Schreter, F. Fogelman-Soulie, & L. Steels (Eds.), *Connectionism in Perspective*. Amsterdam, Netherlands: Elsevier (North-Holland).

Rosenbloom, P. S. & Aasman, J. (1990). Knowledge level and inductive uses of chunking (EBL). *Proceedings of the Eighth National Conference on Artificial Intelligence*. Boston, MA: AAAI, MIT Press.

Rosenbloom, P. S., Laird, J. E., Newell, A., & McCarl, R. (1991). A preliminary analysis of the Soar architecture as a basis for general intelligence. *Artificial Intelligence, 47*, 289-325.

Rosenbloom, P. S., Hirsh, H., Cohen, W. W., & Smith, B. D. (1993). Two frameworks for integrating knowledge in induction. K. Krishen (Ed.), *Seventh Annual Workshop on Space Operations, Applications, and Research (SOAR '93)*. Houston, TX: Space Technology Interdependency Group, NASA Conference Publication 3240.

Rosenbloom, P. S., Laird, J. E., & Newell, A. (Eds.). (1993). *The Soar Papers: Research on Integrated Intelligence*. Cambridge, MA: MIT Press.

Rosenbloom, P. S., Newell, A., & Laird, J. E. (1991). Towards the knowledge level in Soar: The role of the architecture in the use of knowledge. In K. VanLehn (Ed.), *Architectures for Intelligence*. Hillsdale, NJ: Erlbaum.

Simon, H. A., & Lea, G. (1974). Problem solving and rule induction: A unified view. In L. W. Gregg (Ed.), *Knowledge and Cognition*. Potomac, MD: Erlbaum.

Smith, B. D. (December 1995). *Induction as Knowledge Integration*. Doctoral dissertation, University of Southern California,

Smith, B. D. & Rosenbloom, P. S. (1990). Incremental Non-Backtracking Focusing: A polynomially bounded generalization algorithm for version spaces. *Proceedings of the Eighth National Conference on Artificial Intelligence*. Boston, MA: AAAI, MIT Press.

Tambe, M. & Rosenbloom, P. S. (November 1988). *Eliminating Expensive Chunks* (Tech. Rep. #88-189). Department of Computer Science, Carnegie-Mellon University.

Tambe, M. & Rosenbloom, P. S. (1989). Eliminating expensive chunks by restricting expressiveness. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*. Detroit, MI: IJCAII.

Tambe, M. & Rosenbloom, P. S. (1990). A framework for investigating production system formulations with polynomially bounded match. *Proceedings of the Eighth National Conference on Artificial Intelligence*. Boston, MA: AAAI, MIT Press.

Tambe, M. & Rosenbloom, P. S. (1994). Investigating production system representations for non-combinatorial match. *Artificial Intelligence*, *68*, 155-199.

Tambe, M., Newell, A., & Rosenbloom, P. S. (1990). The problem of expensive chunks and its solution by restricting expressiveness. *Machine Learning*, *5*, 299-348.

5. Publications Specifically Supported Under this Effort NCC2-538

Golding, A. R. (October 1991). *Pronouncing Names by a Combination of Rule-Based and Case-Based Reasoning*. Doctoral dissertation, Stanford University,

Golding, A. R. & Rosenbloom, P. S. (1989). Combining Analytical and Similarity-Based CBR. *Proceedings: Case-Based Reasoning Workshop*. Pensacola Beach, FL.

Golding, A., & Rosenbloom, P. S. (1991). Improving rule-based systems through case-based reasoning. *Proceedings of the Ninth National Conference on Artificial Intelligence*. Anaheim, CA: AAAI, MIT Press.

Golding, A. & Rosenbloom, P. S. (1993). A comparison of Anapron with seven other name-pronunciation systems. *Journal of the American Voice I/O Society*, *14*, 1-21.

Golding, A. R. & Rosenbloom, P. S. (1994). The evaluation of Anapron: A case study in evaluating a case-based system. *Working Notes of the AAAI-94 Workshop on Case-Based Reasoning*. Seattle, WA.

Golding, A. R., & Rosenbloom, P. S. (1996). Improving accuracy by combining rule-based and case-based reasoning. *Artificial Intelligence*, . In press.

Laird, J. E., Hucka, M., Huffman, S. B., & Rosenbloom, P. S. (1991). An Analysis of Soar as an Integrated Architecture. *SIGART Bulletin*, *2*, 98-103.

Rosenbloom, P. S. (1989). A symbolic goal-oriented perspective on connectionism and Soar. In R. Pfeifer, Z. Schreter, F. Fogelman-Soulie, & L. Steels (Eds.), *Connectionism in Perspective*. Amsterdam, Netherlands: Elsevier (North-Holland).

Rosenbloom, P. S. & Aasman, J. (1990). Knowledge level and inductive uses of chunking (EBL). *Proceedings of the Eighth National Conference on Artificial Intelligence*. Boston, MA: AAAI, MIT Press.

Rosenbloom, P. S., Hirsh, H., Cohen, W. W., & Smith, B. D. (1993). Two frameworks for integrating knowledge in induction. K. Krishen (Ed.), *Seventh Annual Workshop on Space*

Operations, Applications, and Research (SOAR '93). Houston, TX: Space Technology Interdependency Group, NASA Conference Publication 3240.

Rosenbloom, P. S., Newell, A., & Laird, J. E. (1991). Towards the knowledge level in Soar: The role of the architecture in the use of knowledge. In K. VanLehn (Ed.), *Architectures for Intelligence*. Hillsdale, NJ: Erlbaum.

Smith, B. D. (December 1995). *Induction as Knowledge Integration.* Doctoral dissertation, University of Southern California,

Smith, B. D. & Rosenbloom, P. S. (1990). Incremental Non-Backtracking Focusing: A polynomially bounded generalization algorithm for version spaces. *Proceedings of the Eighth National Conference on Artificial Intelligence.* Boston, MA: AAAI, MIT Press.

Tambe, M. & Rosenbloom, P. S. (November 1988). *Eliminating Expensive Chunks* (Tech. Rep. #88-189). Department of Computer Science, Carnegie-Mellon University.

Tambe, M. & Rosenbloom, P. S. (1989). Eliminating expensive chunks by restricting expressiveness. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*. Detroit, MI: IJCAII.

Tambe, M. & Rosenbloom, P. S. (1990). A framework for investigating production system formulations with polynomially bounded match. *Proceedings of the Eighth National Conference on Artificial Intelligence*. Boston, MA: AAAI, MIT Press.

Tambe, M. & Rosenbloom, P. S. (1994). Investigating production system representations for non-combinatorial match. *Artificial Intelligence*, *68*, 155-199.

Tambe, M., Newell, A., & Rosenbloom, P. S. (1990). The problem of expensive chunks and its solution by restricting expressiveness. *Machine Learning*, *5*, 299-348.